11/12/2024

NotaDomus

Dossier de Conception

par TeapotStudio

S3.01: Création d'une application web pour valoriser le patrimoine culturel français

Equipe 16 :
Emilie DUBIEF, Flavien RIONDET,
Aline ROSTAGNAT, Maxime RASTELLI,
Eddy FRANCOU, Nils RAYOT, Ali KUŞ

Sommaire

Contexte

En tant qu'équipe attachée à la richesse du patrimoine français, nous avons constaté, grâce à un questionnaire d'étude réalisé en amont, que le label *Maison d'Illustres* n'était pas suffisamment reconnu à sa juste valeur. C'est pourquoi nous avons décidé, en groupe de sept personnes, de développer une application web dédiée à la valorisation du patrimoine culturel de la France. Après avoir défini le cadrage de notre projet, nous avons entamé la phase de conception dont les détails sont exposés dans ce document. Par ailleurs, nos progrès pour les phases à venir sont présentés en annexe.

Contraintes et risques

Après notre phase de cadrage, nous sommes revenu sur les notions de contraintes et de risques ainsi que sur une analyse de leur évolution. La matrice de criticité avant mitigation est disponible ici. La plupart des risques voient leur criticité nettement baisser après avoir pris en compte les différentes mitigations comme décrit ici. En particulier, le risque consistant à faire un mauvais choix relatif au langage est largement réduit en prévoyant, sans s'y limiter, des options alternatives.

Besoins

Refonte des besoins non-fonctionnels

Après les retours de notre première présentation orale, nous avons remanié nos besoins non-fonctionnels. La conformité légale n'étant pas un besoin mais une contrainte, nous l'avons enlevée pour la remplacer par les principes du RGPD. Cet ajout nous a permis de bien identifier leur impact sur notre projet. Nous avons donc ajouté le tableau des besoins non-fonctionnels dans l'annexe du document que vous pouvez retrouver ici.

Refonte des besoins fonctionnels

Les retours de la première phase nous ont aussi amené à redéfinir certains besoins fonctionnels. Nos précédents besoins n'étaient pas assez développés, nous en avons donc rajouté et découpé certains. Nous avons ajouté le tableau des besoins fonctionnels dans l'annexe du document que vous pouvez retrouver ici.

Modélisation des besoins

Diagramme des cas d'utilisation

Afin de mieux développer nos cas d'utilisations, nous avons rédigé des scénarios que nous avons placés en annexe. Nous avons fait des scénarios nominaux qui traduisent un comportement classique de nos cas d'utilisations. De plus, nous avons rédigé certains scénarios alternatifs notamment pour illustrer les erreurs de saisie que pourraient faire nos utilisateurs.

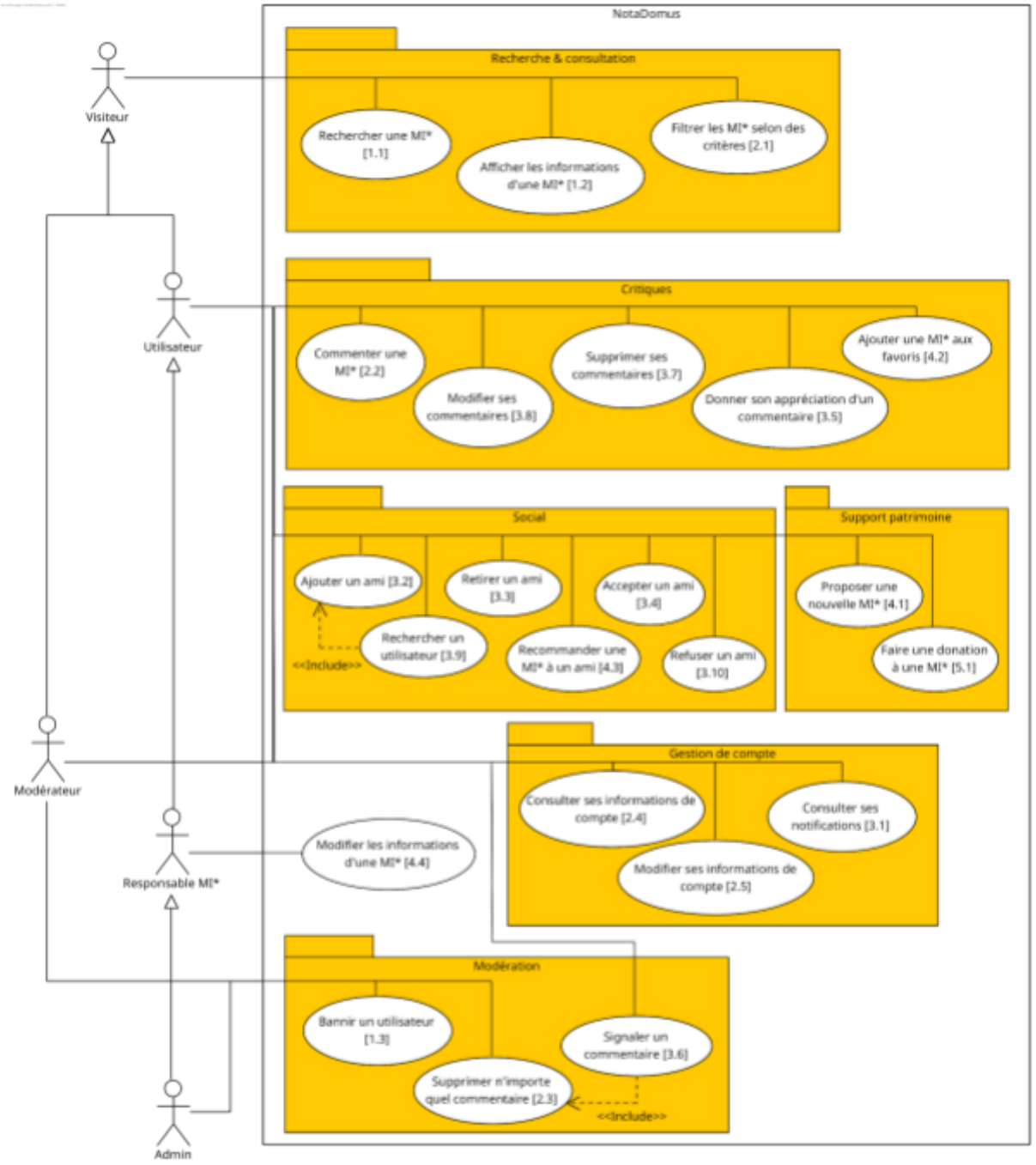
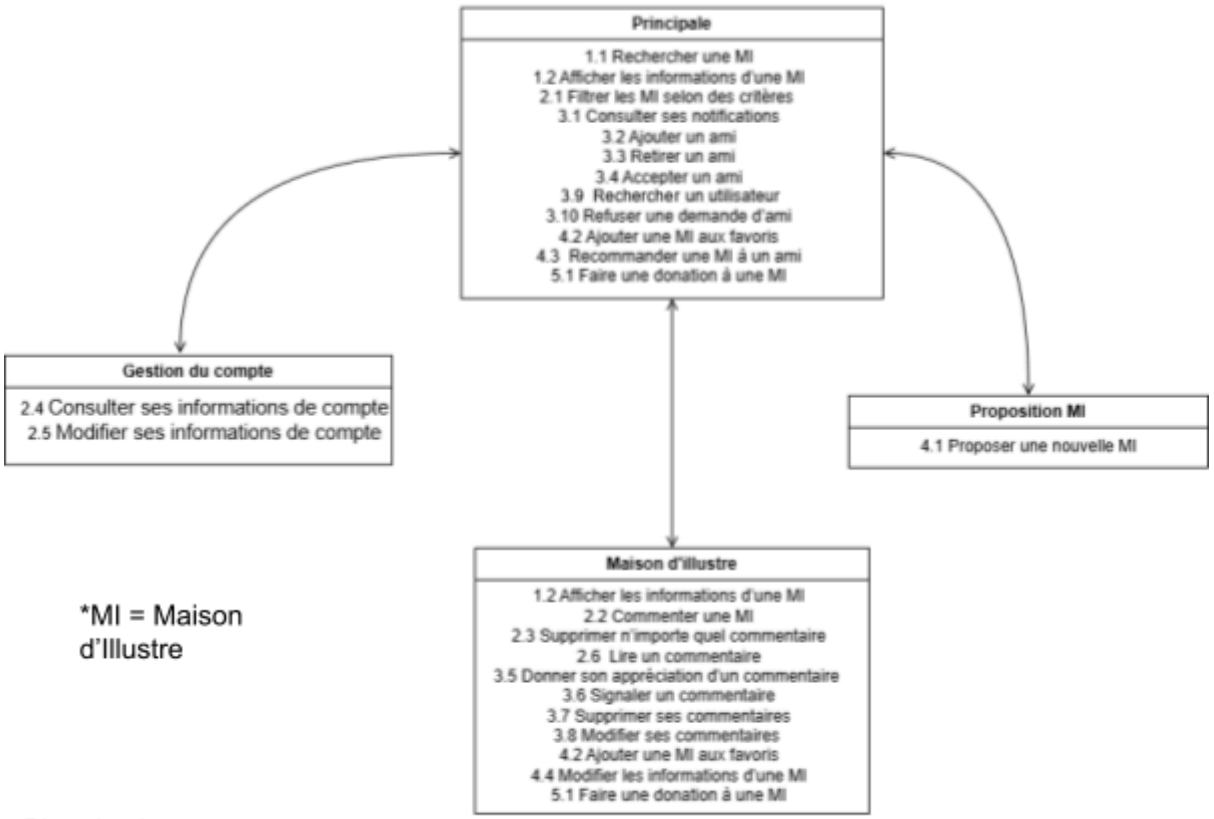


Diagramme Cas D'utilisation

Plan du site

Afin de bien visualiser où se trouvera chaque fonctionnalité de notre application, nous avons réalisé un plan d'organisation des pages. Sur celui-ci, chaque page de l'application contient tous les cas d'utilisation auxquels elle répond, ainsi que la navigation entre les différentes pages.

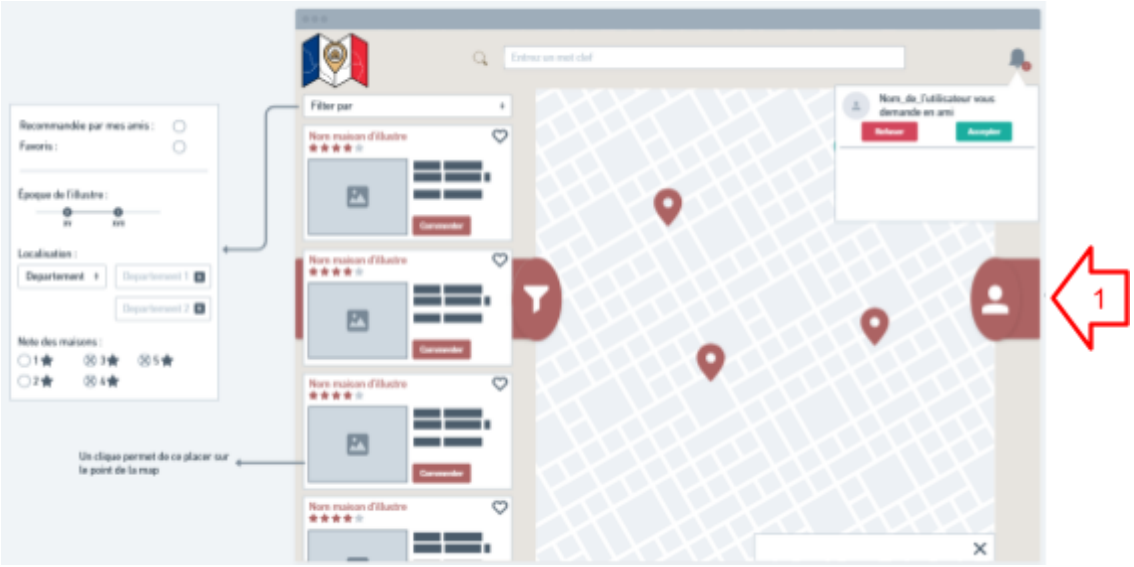


Plan du site

Maquettes

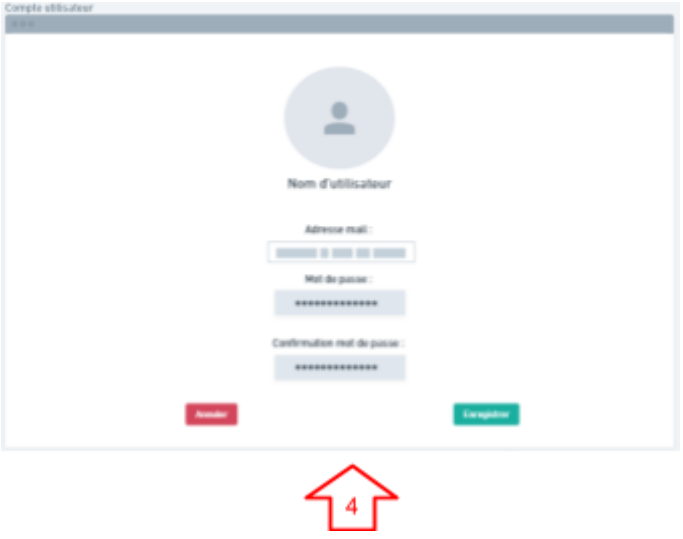
Interfaces

La page principale est composée de l'en-tête avec le logo de l'application qui redirige vers la page principale, une barre de navigation permettant de rechercher une maison selon des mots clés (nom d'illustre, oeuvres) ainsi qu'un icône de cloche pour les notifications. La partie centrale de cette page est la carte interactive comprenant la localisation des maisons d'illustre et l'affichage en pop-up des informations de celles-ci. Un bouton "Plus d'info" permet d'ouvrir une nouvelle page regroupant toutes les informations de la maison. Un pop-up affichant la mascotte de l'application permet à l'utilisateur de pouvoir proposer une nouvelle maison à ajouter. Enfin sur les côtés de la page, se trouvent deux boutons permettant d'ouvrir les filtres et la section communautaire de l'utilisateur.



Lorsque la partie filtres est ouverte, elle se décompose en deux parties. Premièrement la partie "Filtrer par" qui permet de filtrer les maisons selon les recommandations des amis, les favoris de l'utilisateur, l'époque de l'illustre ainsi que la localisation et la note de la maison. Les maisons ne correspondant pas aux filtres n'apparaîtront pas sur la carte. La seconde partie des filtres est composée de plusieurs petits encadrés affichant quelques informations sur les maisons visibles sur la carte. Cliquer dessus permet de se placer sur le point de la carte correspondant à la maison.

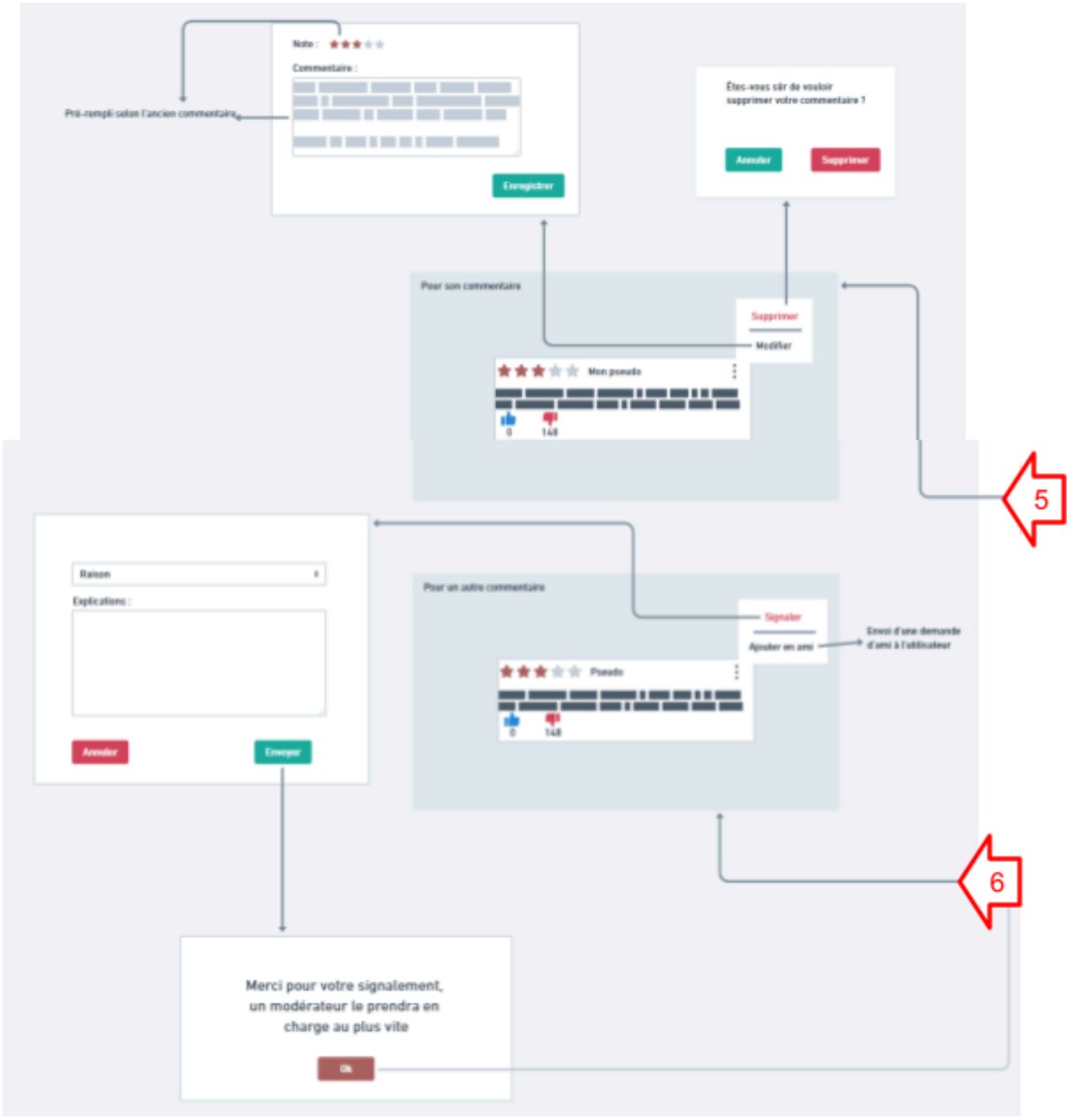
Lorsque l'utilisateur reçoit une notification, celui-ci clique sur le pictogramme des notifications pour les afficher en pop-up.



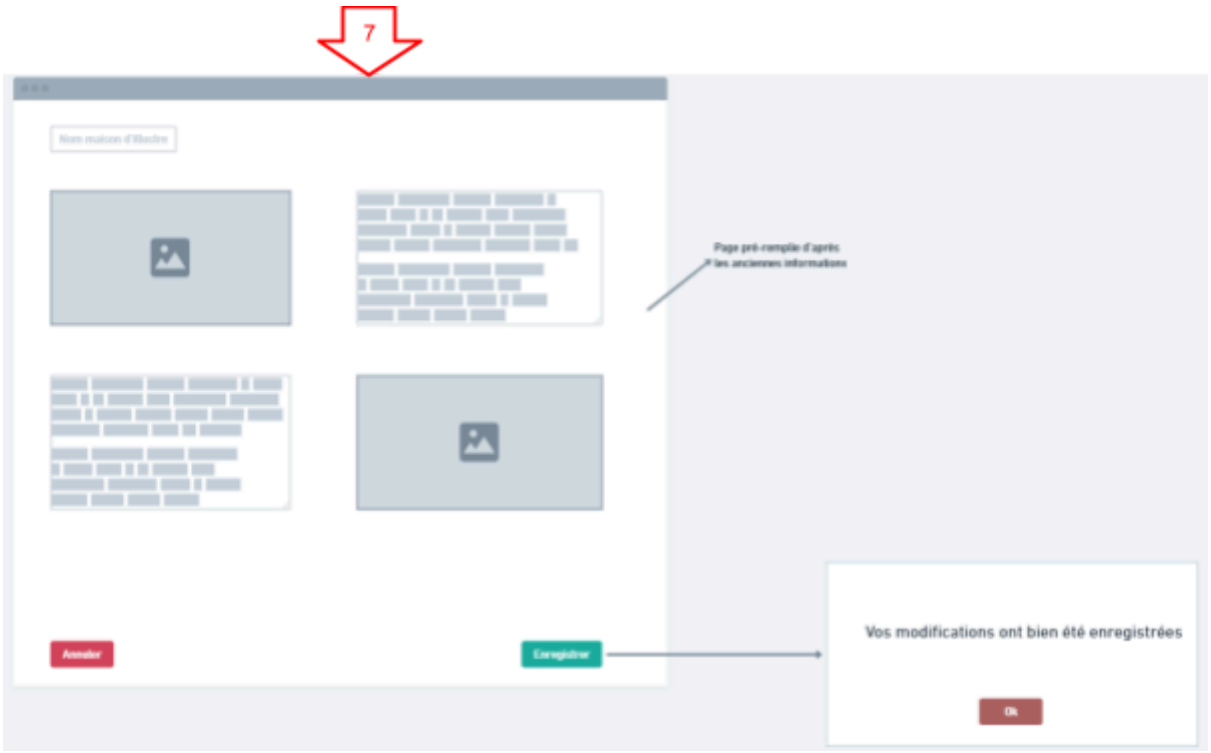
Lorsque l'utilisateur ouvre la partie communautaire, il a accès aux informations de son compte en cliquant sur le bouton "Modifier le compte" qui ouvre une nouvelle page. Cette page, pré-remplie, affiche la photo de profil, le pseudonyme, le mail et le mot de passe de l'utilisateur de manière censurée. Les amis de l'utilisateur sont affichés avec leur photo de profil et leur pseudonyme. L'utilisateur peut ajouter de nouveaux amis en cliquant sur le bouton "Ajouter un ami" et en écrivant le nom de l'utilisateur à ajouter. Si l'utilisateur clique sur un ami, il a la possibilité de le supprimer, un pop-up s'ouvrira afin d'avertir l'utilisateur du caractère irréversible de son acte (échange du code couleur, le bouton "Retirer" devient rouge et le bouton "Annuler" vert). Enfin, l'utilisateur a accès aux derniers commentaires postés par ses amis.

Cette page affiche la totalité des informations de la maison. L'utilisateur peut ajouter celle-ci à sa liste de favoris en cliquant sur l'icône coeur et la recommander à ses amis avec le bouton "⋯" puis "Recommander". Le gérant de la maison d'illustre aura la possibilité de modifier les informations de celle-ci en appuyant sur le bouton "⋯" puis "Modifier les informations". Le bouton de donation permet de récupérer les coordonnées bancaires de l'utilisateur afin de faire un don à la maison. La section commentaires est partagée en deux :

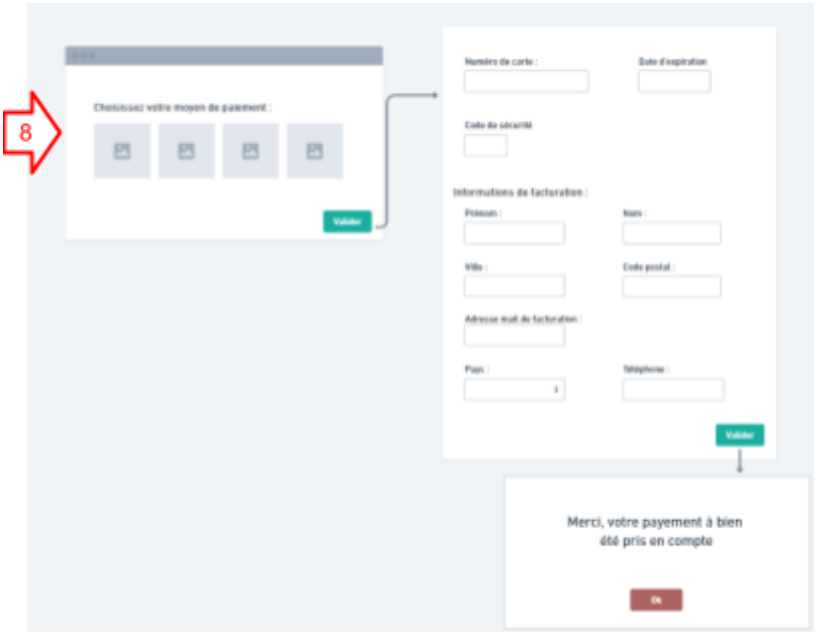
d’une part les commentaires des amis avec la possibilité d’aimer ou non leurs commentaires. L’utilisateur peut poster un nouveau commentaire en remplissant le formulaire de commentaire. La deuxième partie est composée d’autres commentaires.



Un utilisateur peut modifier son commentaire en appuyant sur le bouton “⋯” puis “Modifier” qui ouvrira un pop-up. Le commentaire modifié possédera un label “(Modifié)”. Il peut aussi le supprimer, un pop-up s’ouvrira afin d’avertir l’utilisateur du caractère irréversible de son acte (échange du code couleur, le bouton “Supprimer” devient rouge et le bouton “Annuler” vert). L’utilisateur peut signaler le commentaire d’une autre personne : un pop-up s’ouvrira pour récupérer la raison et une explication du signalement. De plus, il a la possibilité d’ajouter en ami le créateur du commentaire.



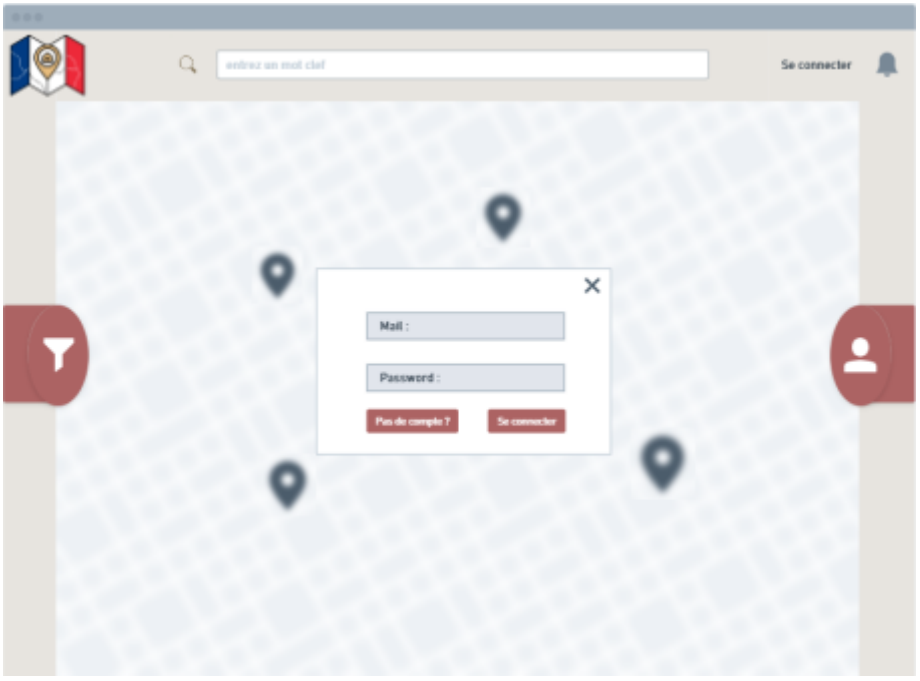
Cette page, visible uniquement par le gérant de la maison, permet de modifier celle-ci à partir d'un formulaire pré-rempli d'après les informations existantes.



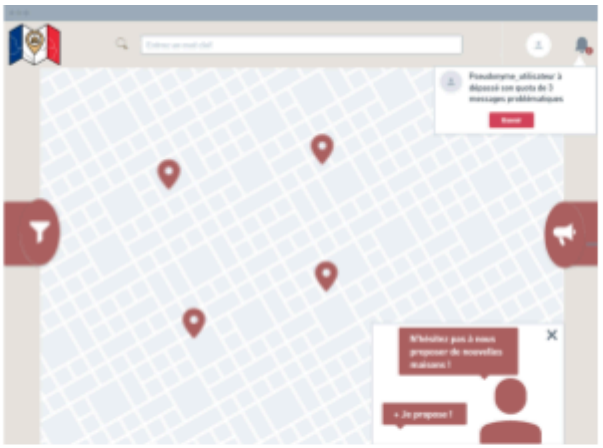
L'utilisateur souhaitant faire une donation devra choisir un moyen de paiement, puis renseigner ses informations bancaires. Celles-ci ne seront pas enregistrées.

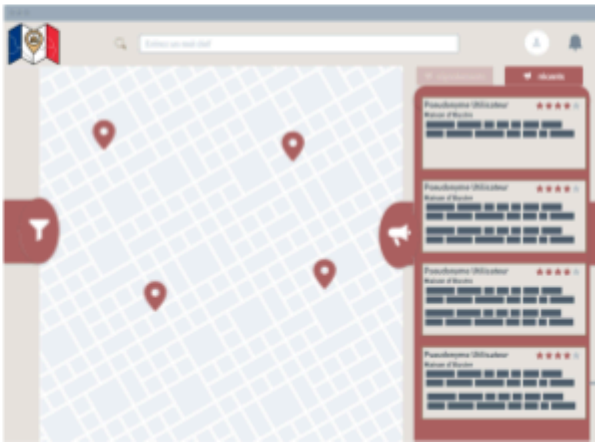
Notre application dispose de plusieurs types de compte, nous avons donc adapté nos maquettes en fonction de ceux-ci.

Pour les visiteurs (les personnes ne possédant pas de compte sur l'application) les fonctionnalités sociales ne seront pas disponibles. A la place, ils verront s'afficher un pop-up leur indiquant qu'ils doivent créer un compte.

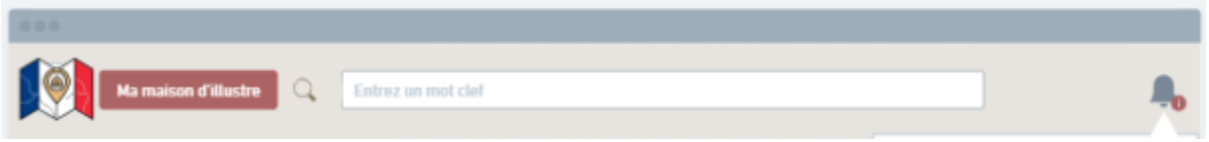


Pour le modérateur, la page principale change. En effet, il ne dispose pas d'amis mais d'un accès plus efficace aux commentaires. Cette interface particulière lui permet d'accéder plus rapidement aux commentaires signalés et aux commentaires les plus récents.





Les gérants de maison d'illustre disposent d'un bouton leur permettant d'accéder rapidement à leur maison. Ils peuvent aussi modifier les informations de cette maison.



Après avoir conçu nos interfaces, nous avons réalisé deux évaluations de leur utilisabilité. Dans un premier temps, nous avons réalisé une analyse heuristique basée sur les critères de Bastien Scapin et sur nos personas que nous avons détaillé ci-dessous. Ensuite, nous avons fait une balade cognitive qui est relatée en annexe.

Analyse heuristique

Valentine Gilmert, étudiante en master d'histoire, souhaite partager sa passion pour les maisons d'illustres et rencontrer des personnes partageant les mêmes centres d'intérêt. Pour répondre à ces besoins, notre interface favorise l'aspect social en rendant accessible facilement les fonctionnalités de partage depuis la page principale de l'application. Le guidage, avec des mécanismes d'aide dans les formulaires de commentaires, facilite les interactions et le partage d'avis. Le regroupement des informations dans des sections spécifiques, comme les commentaires ou les interactions sociales, permet aux utilisateurs ayant le même profil de trouver facilement les fonctionnalités qui l'intéressent. La flexibilité de la plateforme, offrant l'accès aux maisons via une carte interactive ou une barre de recherche, répond à leur curiosité et à leur envie de découverte. Un tutoriel peut également les guider dans l'utilisation des différentes fonctionnalités sociales et d'exploration, renforçant ainsi son expérience utilisateur.

Ghislain Bernard, père de deux jeunes filles, cherche à enrichir leurs connaissances tout en accédant aux informations de manière simple et rapide. Notre interface répond à ses attentes grâce à sa charge de travail, la page principale épurée permet de trouver rapidement les informations sans être submergé par des éléments inutiles. Les retours immédiats, tels que les boutons qui se mettent en surbrillance, facilitent la navigation et rassurent les utilisateurs ayant le même profil que Ghislain sur les actions qu'ils entreprennent. Pour la signifiance des codes et des nominations, une protection contre les erreurs, avec des visuels clairs et des boutons explicites, limite les confusions. Cela est essentiel pour quelqu'un qui veut gagner du temps. De plus, le regroupement des filtres et de la carte centralisée simplifie la localisation des maisons d'illustres adaptées aux intérêts de ses filles.

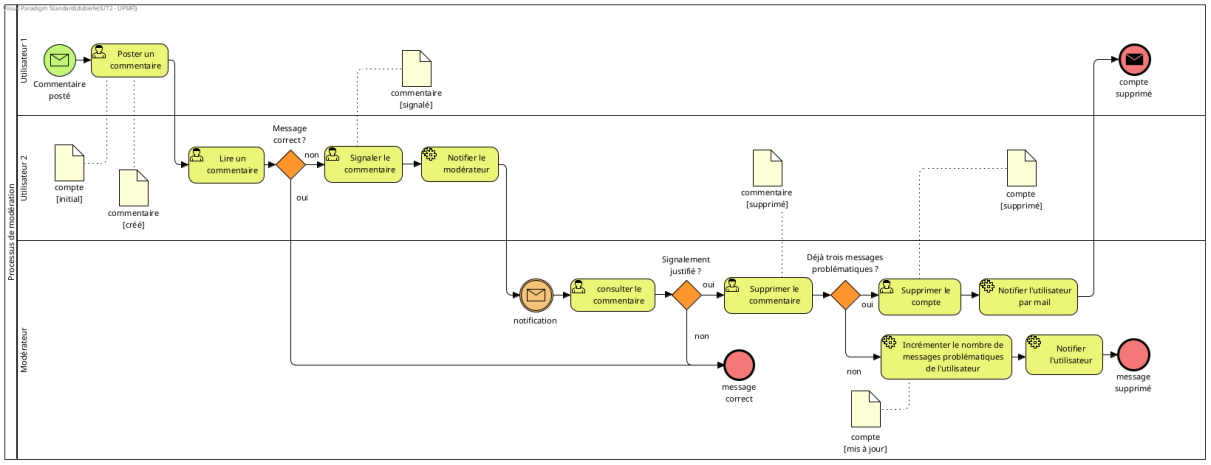
John Smith, homme d'affaires anglais, apprécie de pouvoir explorer le patrimoine français et d'accéder à des avis localisés sur des lieux précis. Notre interface lui permet une navigation efficace grâce au regroupement des informations avec les détails des maisons d'illustres et leurs commentaires. La flexibilité de l'application, qui permet de changer la langue, s'adapte à son profil international. Les retours immédiats via des boutons et logos interactifs en surbrillance améliorent son expérience de navigation en rendant les actions intuitives. Enfin, l'homogénéité visuelle et textuelle de l'application renforce la crédibilité et facilite l'utilisation pour un utilisateur professionnel comme John.

Henriette Delacours, gérante du Château d'Hautefort, cherche à valoriser son héritage et à rendre son château plus visible et reconnu. L'interface a été pensée pour faciliter la compréhension grâce à des icônes adaptées et explicites pour des fonctionnalités comme les filtres ou les notifications, ce qui correspond à son besoin de simplicité. La flexibilité de l'application permet de répondre aux attentes d'un public varié, essentiel pour toucher un maximum de visiteurs potentiels. La protection contre les erreurs est assurée par des messages clairs qui guident Henriette dans l'utilisation de la plateforme, notamment lorsqu'elle remplit des informations sur son château. Un tutoriel, en plus d'une interface claire et intuitive, peut également l'aider à mettre en avant son patrimoine auprès des utilisateurs de l'application.

Diagramme BPMN

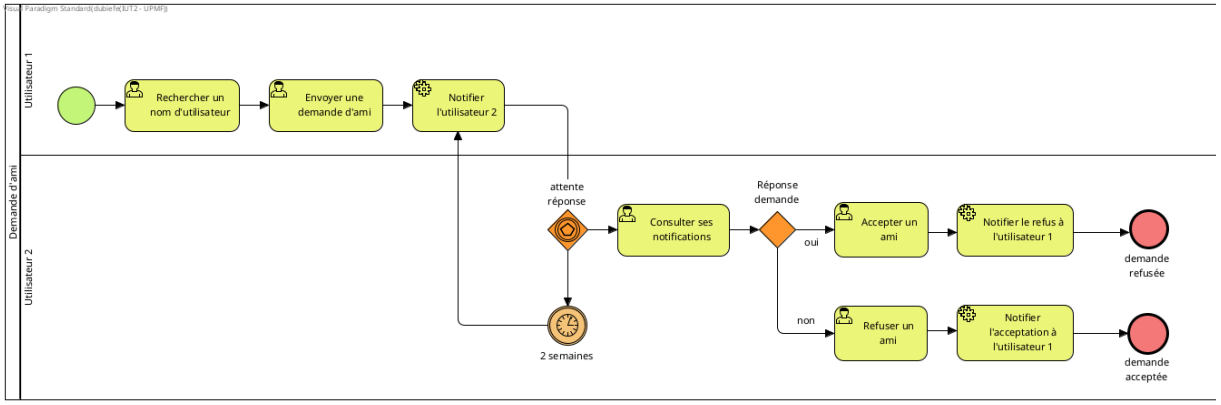
Afin de mieux visualiser certains de nos processus, nous avons réalisé deux diagrammes BPMN.

Le premier représente le processus de modération. Nous avons identifié trois acteurs internes: deux utilisateurs (nous avons pris en compte le fait qu'un utilisateur ne va pas se reporter lui-même) et un modérateur.



Ce processus nous a permis d'identifier deux objets qui seront des classes dans le diagramme de classes: un compte et un commentaire.

Le deuxième représente le processus de demande d'ami. Nous avons identifié deux acteurs: deux utilisateurs, un à l'origine de la demande et l'autre qui la reçoit.



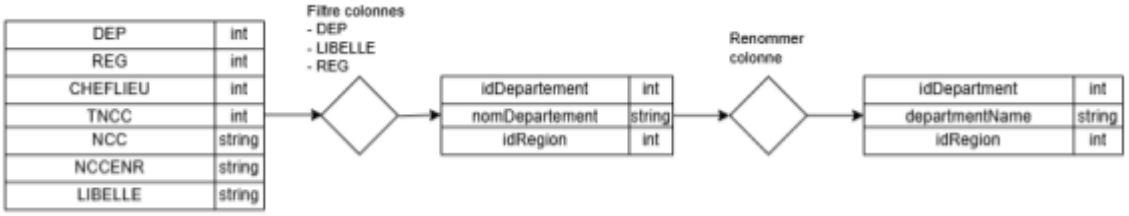
Ces deux processus nous ont aussi permis de nous rendre compte de la nécessité de mettre en place un système de notifications au sein de l'application pour faciliter la communication entre les utilisateurs.

Modèle de données

Pour constituer la base de données nécessaire à notre application web, nous avons identifié les jeux de données brutes fournies sur le site du gouvernement qui nous sont utiles. Nous en avons sélectionné trois: la base des maisons d'illustres françaises, la base des régions françaises et celle des départements. Ces deux dernières nous seront utiles notamment pour les filtres de recherche de la carte interactive.

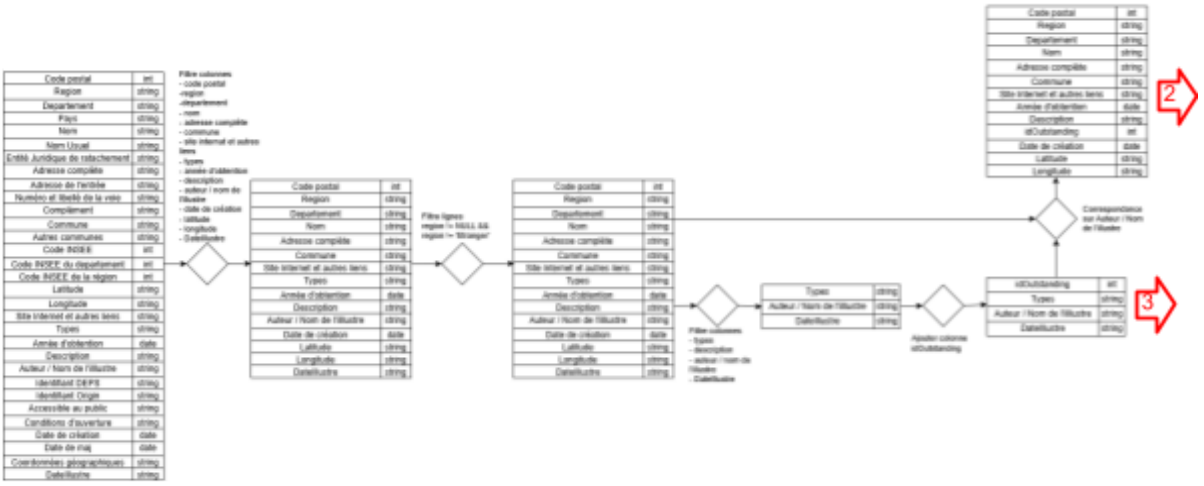
Flux de transformations

Afin d'extraire au mieux les données nécessaires, nous avons réalisé plusieurs schéma de flux de transformations. Nous réaliserons ces transformations avant la phase de réalisation grâce au logiciel Talend. Les deux premiers flux concernent les jeux de données des régions et départements.



Flux de transformations régions et départements

Le prochain flux de transformations concerne le jeu de données des maisons d'illustre. Ce diagramme étant trop grand, nous l'avons décomposé en trois parties.



Flux de transformations maisons d'illustre - Partie 1

La deuxième partie de ce flux de transformation illustre la création de la relation contenant toutes les informations concernant les maisons d'illustre.

Flux de transformations maisons d'illustre - Partie2

La troisième partie de ce flux représente la création de la relation contenant toutes les informations concernant les illustres.

Flux de transformations maisons d'illustre - Partie 3

Schéma Entités Association

A partir des flux de transformations réalisés, nous avons pu identifier quatre relations pour notre base de données: les maisons, les illustres, les régions et les départements. Nous avons ensuite enrichi notre schéma de base de données avec des relations supplémentaires. Nous avons choisi d'utiliser des noms de tables et d'attributs en anglais pour que cela soit cohérent avec la syntaxe des langages de programmation et de nos conventions de nommage. En annexe, nous avons aussi mis le schéma logique relationnel et le fichier create.sql qui correspondent au schéma Entités Association qui suit.

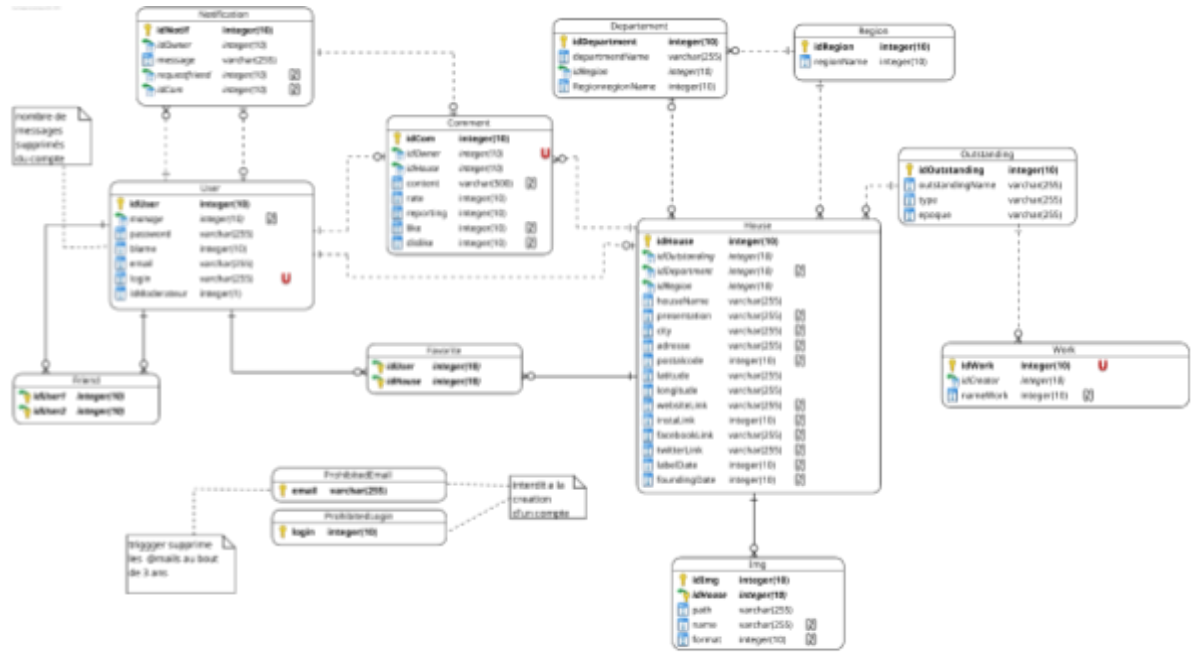


Schéma Entités Association

Diagrammes de classes

Après avoir identifié la base de données de notre application, nous avons construit deux diagrammes de classes. Le premier ne contient que les attributs de chaque classe. Le second contient en plus les méthodes. Nous avons aussi mis ici, en annexe, des diagrammes d'objet pour concrétiser ces diagrammes de classes.

Diagramme simplifié

Ce diagramme sert essentiellement à représenter sous forme orientée objet le schéma de données (avoir une structure de données similaire, mais accessible pendant l'exécution). Il ajoute néanmoins une distinction entre les différents User de l'application. Et sépare la table House pour la simplifier et rendre plus simple l'accès aux données géographiques des maisons lors de leur affichage sur la carte.

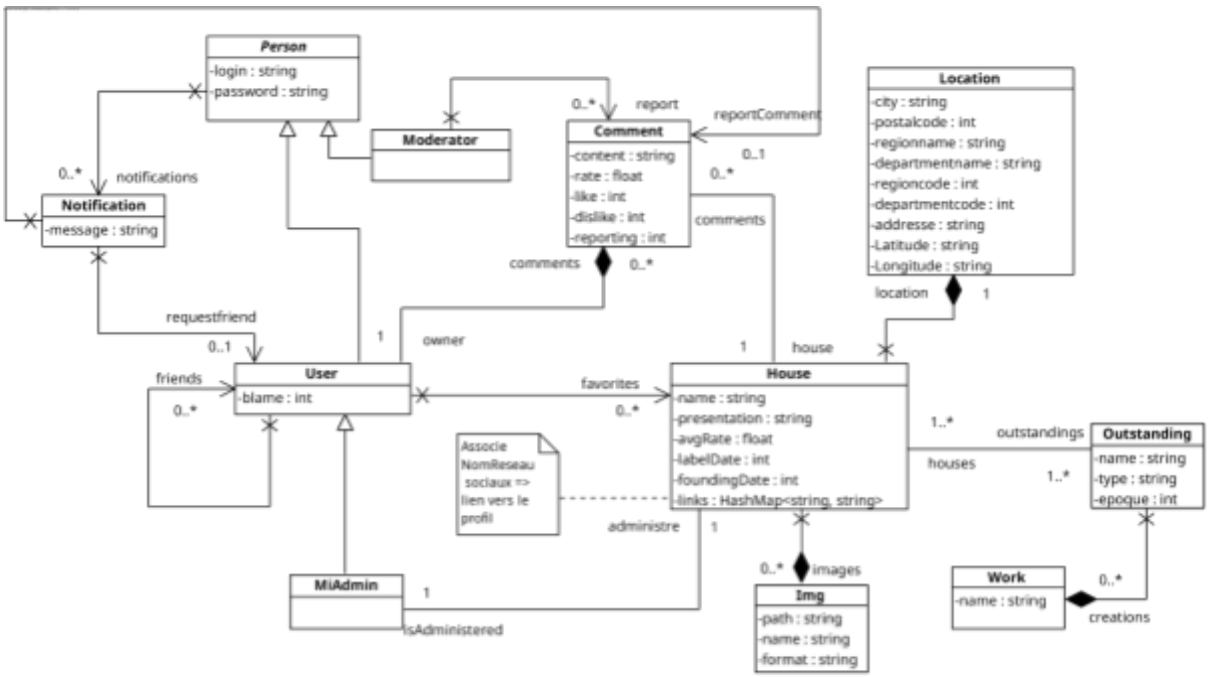


Diagramme de Classe simple

Diagramme complet

Ce diagramme permet de montrer plus en détail les méthodes des différentes classes. Nous n'avons pas renseigné les getters et les setters qui sont présents pour chaque attributs des classes.

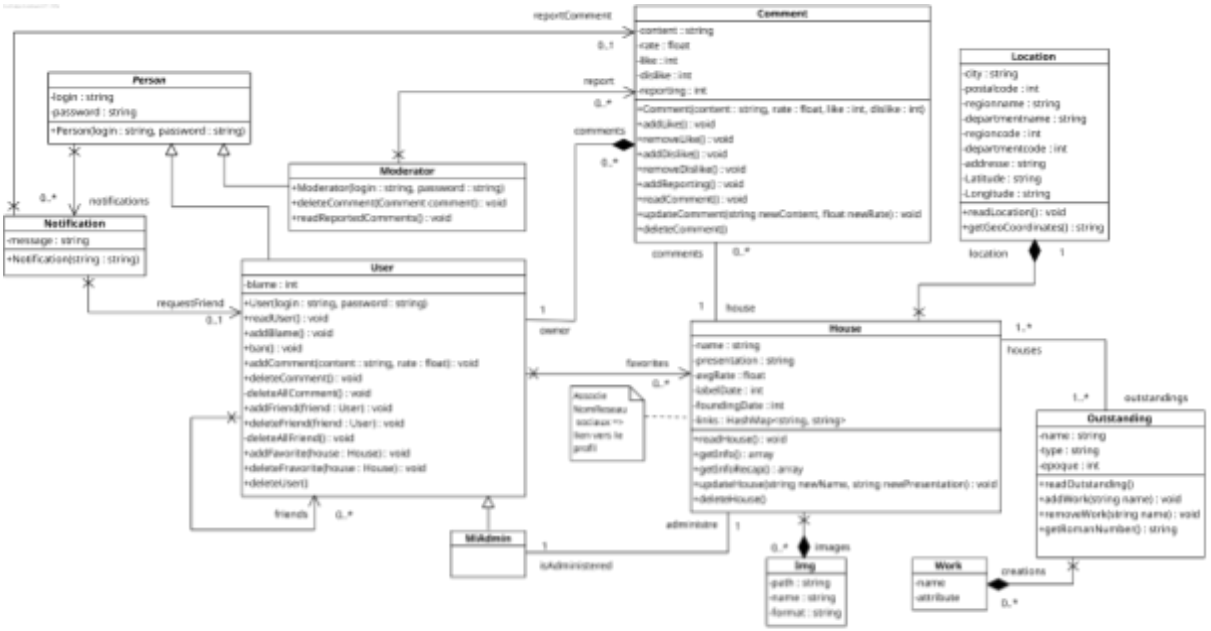


Diagramme de Classe Complet

Diagrammes de séquences

Ce diagramme permet de nous projeter dans l'utilisation de certaines méthodes afin d'afficher chacune des classes.

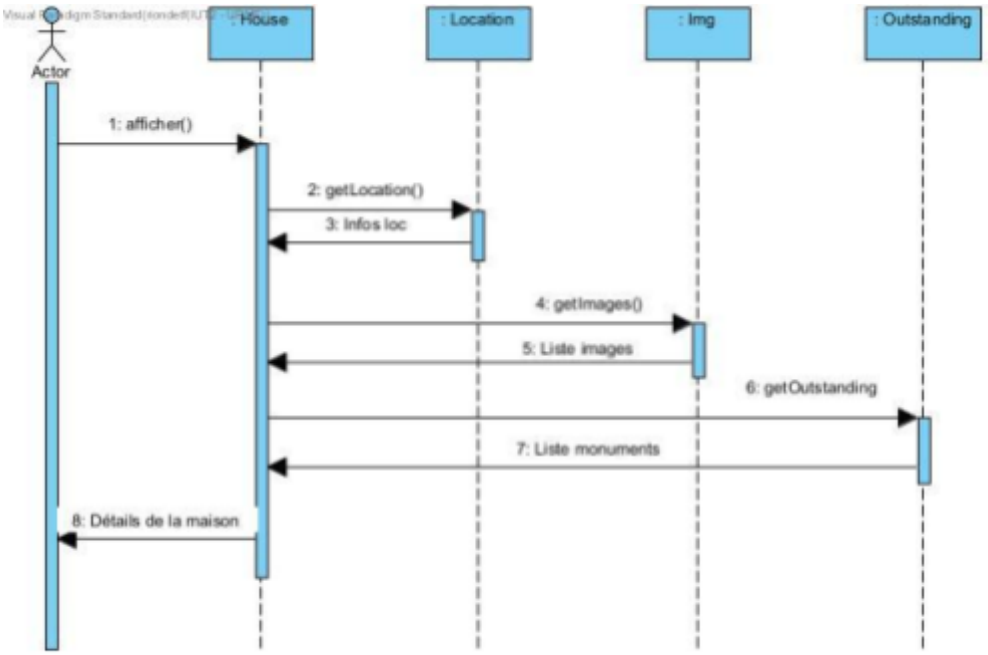


Diagramme de séquences - afficher()

Réalisation

Etude de faisabilité

Analyse des exigences techniques

Spécifications techniques (fonctionnalités principales) :

- Carte interactive.
- Recherche et tris par pertinence des résultats.
- Gestion des commentaires.
- Modèle serveur/client web.
- Compatibilité avec les principaux navigateurs (Chrome, Firefox, Edge, etc.).

Normes et standards :

- L'application doit suivre une architecture MVC (modèle étudié en cours).
- Respect des consignes du RGPD (recueil du consentement pour les données personnelles collectées). Une bannière de cookies ou un formulaire de consentement devra être prévu.

Technologies et outils nécessaires

Choix technologiques (détaillés en dessous) :

- Carte : API de Google Maps (avec possibilité d'utiliser OpenStreetMap comme alternative).
- Langage serveur : PHP.
- Langage client : HTML/CSS/JavaScript.

- Serveur web : Debian 12, Apache2.
- Base de données : PostgreSQL (sur serveur Debian 12).

Compatibilité des systèmes :
- mod_php : Module Apache permettant de gérer PHP.
- PDO_PGSQL : Extension standard PHP offrant une interface orientée objet pour accéder aux bases de données PostgreSQL.

Plans pour combler les lacunes :
- Suivre des tutoriels officiels et pratiques pour maîtriser l'API Google Maps.
- Approfondir JavaScript pour manipuler des APIs et gérer des interactions dynamiques.

Infrastructures

Capacité de mise à l'échelle :
- Si la base de données ou le code requièrent plus d'espace de stockage que prévu, l'espace de stockage des serveurs virtuels peut être augmenté.

Maintenance et support :
- Utilisation de la version stable de Debian 12 pour garantir la fiabilité.
- Aucune mise à jour majeure n'est prévue après la fin du projet.

Risques et solutions

Risque : Dépassement des quotas ou des coûts liés à l'API Google Maps.
Solution : Avoir une alternative (ex. OpenStreetMap) et surveiller les statistiques d'usage.

Risque : Problèmes de compatibilité avec certains navigateurs.
Solution : Effectuer des tests cross-browser (Chrome, Firefox, Edge, Safari). Utiliser des technologies générales (éviter par exemple -webkit-...)

Technologies utilisées

Afin de mener à bien notre projet, nous avons convenu des technologies à adopter. Pour le développement front-end, nous avons choisi d'utiliser les langages HTML, CSS et JavaScript. Concernant le back-end, nous avons opté pour PHP et SQL. Pour exploiter ces langages, nous utiliserons les éditeurs VScode ou PhpStorm ainsi que le système de gestion de bases de données PostgreSQL. Nous avons fait le choix de ces technologies car elles font partie intégrante de notre formation au BUT, tous les membres de l'équipe sont donc à même de les utiliser selon leur degré d'affinité. De plus, le PHP est spécifiquement conçu pour le back-end Web.

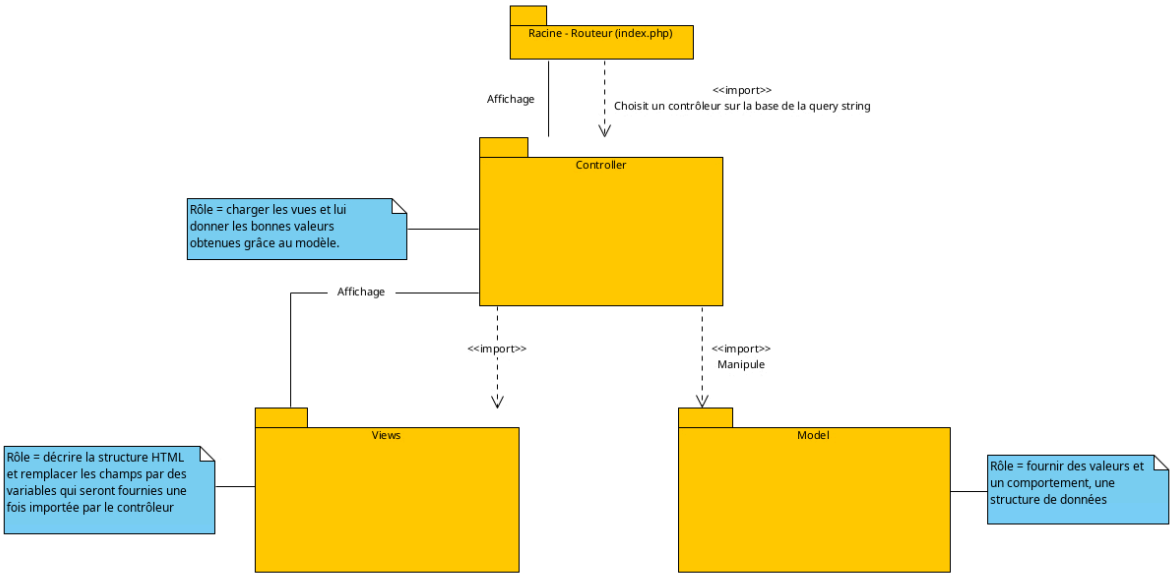
Nous avions aussi pensé à utiliser le langage Java au lieu de PHP, cependant après avoir étudié la question, le fait de devoir utiliser une nouvelle spécification (Jakarta EE) afin de pouvoir le relier aux langages web nous est apparu trop compliqué à mettre en place et coûteux en temps.

Afin d'afficher une carte interactive, nous avons décidé d'utiliser une API. Nous avons d'abord voulu utiliser l'API de Géoportail mais celle-ciest trop limitée en version gratuite et n'est fonctionnelle que pour des tests ou une utilisation personnelle. De plus, elle

ne dispose que d'une carte interactive de la France métropolitaine, nous risquons d'être bloqués pour afficher certaines maisons présentes dans des départements d'outre-mers. Nous avons donc choisis l'API de Google Maps puisqu'elle est gratuite tant que notre quota de requêtes ne dépasse pas 30 000. Enfin, cette API affiche une carte du monde ce qui nous permettra d'afficher les maisons d'illustres des départements d'outre-mers. Nous allons utiliser cette API via JavaScript.

Architecture logicielle

Côté serveur, nous avons opté pour une architecture MVC avec routeur. Cela signifie qu'il y aura un unique point d'entrée à l'application (index.html). Un contrôleur est choisi en tenant compte des informations de la query string. Le contrôleur utilise ainsi le modèle pour réaliser les opérations nécessaires et passer les valeurs aux vues qui affichent simplement les variables. La structure peut donc être résumée ainsi :

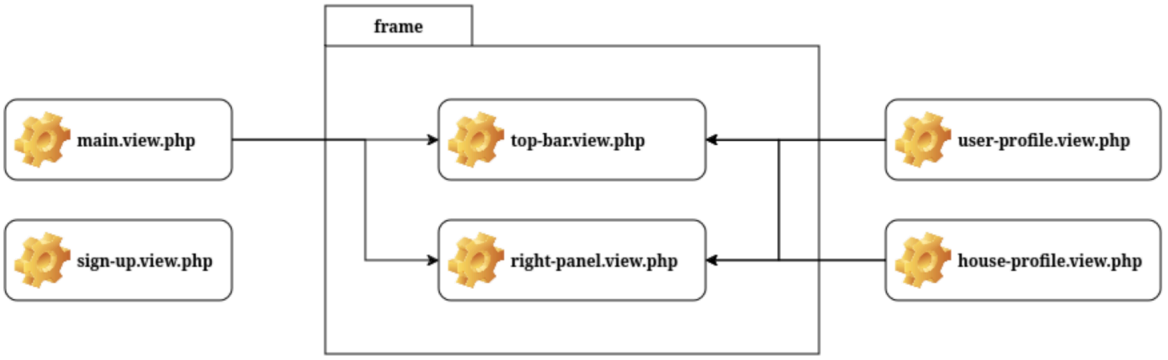


Structure générale de l'application

Le schéma ci-dessous illustre la structure de la partie "vues". Celle-ci est constituée principalement de 4 vues :

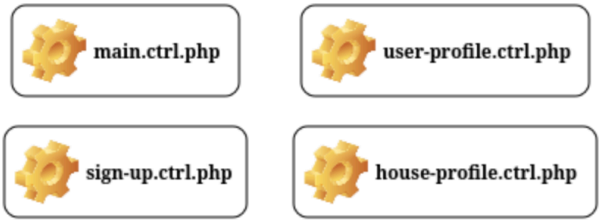
- La vue principale, correspondant à la carte interactive
- La vue d'inscription
- La vue du profil utilisateur
- La vue d'une maison d'illustre

Certaines de ces vues utilisent les mêmes composants qui sont donc factorisés dans le package "frame". Cela inclut le panneau de droite (permettant de consulter sa liste d'amis et les commentaires récents), ainsi que la barre de navigation et le champ de recherche. Les vues principales, profil utilisateur et maison d'illustre utilisent toutes ces composants.



Structure détaillée des vues

La partie “contrôleur” est composée d’un fichier pour chaque vue. Chaque contrôleur charge la vue correspondante.



Les quatre contrôleurs

Côté client, un seul fichier Javascript contenant un ensemble de fonctions sera utilisé pour modifier l’affichage en temps réel selon les actions de l’utilisateur sur une même page.

Mise en place des serveur

Les serveurs ont passé plusieurs tests qui sont cités ci-dessous. Pendant leur création, il a fallu installer Apache sur l’un pour pouvoir mettre en ligne une application web en PHP. Et installer sur l’autre Postgresql qui va gérer nos bases de données. Pour relier les deux, nous avons modifié quelques fichiers de configurations de sorte à ce qu’ils puissent communiquer et s’échanger des informations. Un site web très restreint a été créé pour vérifier si les injections SQL étaient bien bloquées, et assurer l’intégration.

Tests d’intégration :

Objectif : vérifier la base de données en ajoutant des requêtes pour vérifier le bon fonctionnement.

Tests fonctionnels :

Objectif : valider que chaque utilisateur UNIX (membres de l'équipe) puisse se connecter sur le bon compte.
Nous avons créé des utilisateurs sur chaque serveur et attribué à chacun un homedir.

Tests de sécurité :

Objectifs :

- Tester contre des attaques type injection SQL.

- Vérifier que les données sensibles sont hachées.

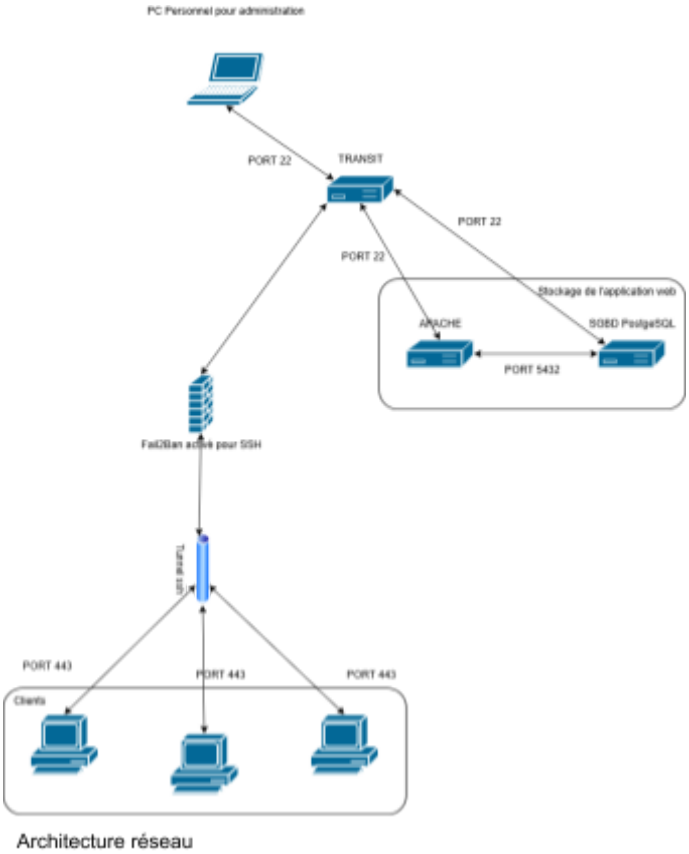
Sur le site de test créé pour tester la base de données, la récupération se fait en PHP et l'entrée est transformée obligatoirement en string.

Tests de compatibilité :

Objectif : vérifier que le serveur fonctionne sur différents systèmes d'exploitation ou navigateurs.

Le site de test créé pour l'occasion tourne effectivement sur plusieurs navigateurs.

Architecture réseau



Architecture réseau

Planification

Lors de cette deuxième itération, nous avons concentré nos efforts sur la conception, en nous appuyant sur le cadrage réalisé lors de la première phase pour poser des bases solides. Cette étape nous a permis d'aborder l'expression et la modélisation des besoins de manière plus approfondie. Nous avons remodelé les besoins fonctionnels et non-fonctionnels, révisé les cas d'utilisation, et produit des livrables clés tels que des maquettes, des diagrammes BPMN, un modèle de données et un plan du site. Ces éléments nous permettent de mieux visualiser l'application et de préparer sa mise en œuvre. Nous avons mené une étude approfondie de faisabilité technique afin de valider les choix technologiques envisagés. Nos tests ont confirmé que les outils sélectionnés répondent aux exigences du projet en termes de performance, compatibilité et évolutivité. Par ailleurs, l'architecture logicielle a été élaborée, structurant les différents modules et leurs interactions,

de plus, l'architecture réseau a été définie pour répondre aux besoins d'hébergement et de communication.

Malgré ces avancées, plusieurs étapes importantes restent à accomplir pour finaliser la phase de réalisation. Le développement des fonctionnalités principales sera également une priorité dans les semaines à venir, en se basant sur les modèles et maquettes définis lors de la conception. La création des environnements de test est un autre point essentiel, afin de garantir la qualité des développements futurs et d'assurer une intégration fluide des différentes fonctionnalités. Enfin, une révision de la planification permettra d'ajuster les étapes restantes en fonction des progrès réalisés et des éventuelles contraintes identifiées.

Annexe

Matrice de criticité avant mitigation

Impact \ Probabilité	1	2	3
1		RC5 RH6	RTechM1 RTechI6
2	RTechI1 RH1 RH3 RH4 RH7 RTemp4	RTemp3 RTechI3 RTechI4	
3	RTemp1 RTemp2 RC6 RH2 RH5 RTechI5	RTechI2	

Matrice de criticité après mitigation

Impact \ Probabilité	1	2	3
1	RTechI1 RH4 RH6 RH7 RTemp4	RC5 RTechI3 RTechM1	RTechI6

2	RTechI5 RH1 RH3 RH5 RC6 RTemp2 RTemp3	RTechI2 RTechI4	
3	RTemp1 RH2		

Tableau des besoins non-fonctionnels

Nom du besoin	Priorité	Description
Licéité, loyauté et transparence de la collecte des données	1	L'utilisateur doit être mis au courant de la collecte de ses données.
Finalités déterminées, explicites et légitimes	1	L'utilisateur doit être mis au courant de la manière dont ces données vont être utilisées.
Collectes et traitements adéquats, pertinents et limites	1	Minimiser la collecte de données personnelles.
Exactitude des données	1	Les données doivent être mises à jour régulièrement.
Conservation limitée des données	1	La durée de conservation doit être connue et appropriée à l'utilisation.
Sécurité	1	Sécurité des données des comptes des utilisateurs
Maintenabilité	2	Le code doit pouvoir être compris rapidement
Ergonomie	2	L'application doit pouvoir être prise en main facilement

Tableau des besoins fonctionnels

ID	Nom du UC	Priorité	Description
1.1	Rechercher une maison d'illustre	1	affiche des points sur la carte
1.2	Afficher les informations d'une maison d'illustre	1	donne le nom de la MI, une photo, une description, les commentaires
1.3	Bannir un utilisateur	1	supprime le compte et donc tous les commentaires
2.1	Filtrer les maisons d'illustres selon des critères	2	affiche une prévisualisation des maisons et des points sur la carte selon les critères de filtre

2.2	Commenter une maison d'illustre	2	ajouter un commentaire et une note à une maison (un commentaire par maison)
2.3	Supprimer n'importe quel commentaire	2	supprimer un commentaire qui n'est pas le nôtre, et qui n'est donc plus visible par les autres utilisateurs
2.4	Consulter ses informations de compte	2	voir son mail, son pseudo, ses points
2.5	Modifier ses informations de compte	2	modifier son adresse mail, son pseudo, son mot de passe
3.1	Consulter ses notifications	3	voir si on a des recommandations, des demandes d'amis, un message offensant supprimé
3.2	Ajouter un ami	3	envoyer une demande d'ami à un utilisateur
3.3	Retirer un ami	3	retirer un utilisateur de sa liste d'ami
3.4	Accepter un ami	3	accepter une demande d'ami d'un utilisateur
3.5	Donner son appréciation d'un commentaire	3	mettre un avis favorable ou défavorable à l'aide de boutons
3.6	Signaler un commentaire	3	envoyer une notification au modérateur
3.7	Supprimer ses commentaires	3	effacer un de ses commentaires
3.8	Modifier ses commentaires	3	modifier sa note ou le contenu texte d'un commentaire
3.9	Rechercher un utilisateur	3	rechercher le pseudonyme d'un utilisateur pour l'ajouter en ami
3.10	Refuser un ami	3	refuser une demande d'ami d'un utilisateur
4.1	Proposer une nouvelle maison d'illustre	4	remplir un formulaire qui s'envoie à une adresse mail dédiée
4.2	Ajouter une maison d'illustre aux favoris	4	ajoute un maison dans les favoris
4.3	Recommander une maison d'illustre à un ami	4	envoie une notification à un ami
4.4	Modifier les informations d'une maison d'illustre	4	changer le texte, les images de la page de la MI
5.1	Faire une donation à une maison d'illustre	5	remplir un formulaire de paiement

Scénarios

Scénarios nominaux

Scénario principal	UC1.1 Rechercher une maison d'illustre
Description	1. L'utilisateur entre un nom de maison, un nom d'illustre ou une œuvre de l'illustre dans la barre de recherche. 2. L'application affiche sur la carte les différents points d'intérêts respectant la recherche.

Scénario principal	UC1.2 Afficher les informations d'une maison d'illustre
Description	1. L'utilisateur appuie sur un point d'intérêt. 2. L'application affiche une prévisualisation de la maison (nom, photo, note, favoris, courte description, bouton "commenter" et bouton "plus d'info"). 3. L'utilisateur appuie sur le bouton "plus d'info". 4. L'application ouvre une nouvelle page regroupant des informations supplémentaires sur la description et les commentaires de la maison.

Scénario principal	UC1.3 Bannir un utilisateur
Description	1. L'application notifie le modérateur qu'un utilisateur à dépasser son quota de 3 messages problématiques supprimés. 2. Le modérateur appuie sur le bouton "Bannir". 3. L'application supprime le compte de l'utilisateur, lui envoie un message pour l'informer de la suppression de son compte et met son adresse mail sur liste noire.

Scénario principal	UC2.1 Filtrer les maisons d'illustres selon des critères
Description	1. L'utilisateur clique sur le bouton des filtres. 2. L'application affiche une choice box des filtres disponibles. 3. L'utilisateur appuie sur la choice box "filtrer par". Il choisit ses filtres selon l'époque de l'illustre, une localisation, les notes des maisons, les recommandations de ses amis et ses favoris puis il valide ses filtres. 4. L'application n'affiche plus que les maisons respectant ces critères.

Scénario principal	UC2.2 Commenter une maison d'illustre
Description	1. L'utilisateur clique sur la maison pour laquelle il veut écrire un commentaire. 2. L'application affiche les informations sur la maison. 3. L'utilisateur appuie sur le bouton "commenter". 4. L'application ouvre un champ d'écriture. 5. L'utilisateur écrit son commentaire et sa note puis valide son entrée. 6. L'application ajoute son commentaire à la liste des commentaires de la maison.

Scénario principal	UC2.3 Supprimer n'importe quel commentaire
Description	1. Le modérateur clique sur le commentaire à supprimer. 2. L'application affiche le bouton "⋯" puis "Supprimer". 3. Le modérateur appuie sur le bouton "Supprimer". 4. L'application supprime le commentaire, incrémente le nombre de messages déplacés de l'utilisateur puis envoie une notification de la confirmation de la suppression du commentaire au modérateur.

Scénario principal	UC2.4 Consulter ses informations de compte
Description	1. L'utilisateur clique sur son image de profil. 2. L'application ouvre une nouvelle page qui affiche son profil (image, pseudonyme, email, mot de passe masqué) avec un bouton "modifier".

Scénario principal	UC2.5 Modifier ses information de compte
Description	1. L'utilisateur clique sur la partie relation de l'application puis sur le bouton "Modifier" en face de son pseudonyme. 2. L'application ouvre une nouvelle page qui affiche son profil en mode modification et affiche ses informations dans des champs de texte. 3. L'utilisateur écrit dans les champs qu'il souhaite modifier et appuie sur le bouton "Enregistrer". 4. L'application met à jour les informations de l'utilisateur et lui envoie une confirmation par notification et par mail.

Scénario principal	UC3.1 Consulter ses notifications
Description	1. L’utilisateur clique sur les notifications. 2. L’application affiche une liste des notifications de l’utilisateur ainsi que leur date d’apparition.

Scénario principal	UC3.2 Ajouter un ami
Description	1. L’utilisateur clique sur le bouton du menu des relations. 2. L’application affiche une section à droite de l'écran avec ses amis, les derniers commentaires de ceux-ci et un bouton “Ajouter un ami”. 3. L’utilisateur appuie sur le bouton “Ajouter un ami”. 4. L’application affiche une barre de recherche demandant d’entrer le nom de l’ami à ajouter et un bouton “Envoyer”. 5. L’utilisateur remplit la barre et appuie sur le bouton “Envoyer”. 6. L’application envoie une notification à l'utilisateur à ajouter en tant qu’ami.

Scénario principal	UC3.3 Retirer un ami
Description	1. L’utilisateur clique sur le bouton du menu des relations. 2. L’application affiche ses amis. 3. L’utilisateur appuie sur l’ami à retirer. 4. L’application affiche un bouton “Retirer l’ami”. 5. L’utilisateur appuie sur le bouton “Retirer l’ami”. 6. L’application supprime l’ami des relations de l’utilisateur. 7. L’application supprime l'utilisateur des relations de l’ancien ami.

Scénario principal	UC3.4 Accepter un ami
Description	1. L’application envoie à l'utilisateur une notification l’informant qu'il a reçu une demande d’ami. 2. L’utilisateur ouvre ses notifications. 3. L’application informe l’utilisateur que tel autre utilisateur le demande en ami et affiche un bouton “Accepter” et un autre “Refuser”. 4. L’utilisateur appuie sur le bouton “Accepter”. 5. L’application ajoute l’ami à la liste des relations de l’utilisateur.

Scénario principal	UC3.5 Donner son appréciation d’un commentaire
Description	1. L’application affiche les commentaires d’une MI avec pour chacun, la possibilité de donner une appréciation positive avec un pouce en l’air ou une négative avec un pouce vers le bas. 2. L’utilisateur appuie sur l’un de ces pouces selon son avis sur le commentaire. 3. L’application comptabilise l'appréciation.

Scénario principal	UC3.6 Signaler un commentaire
Description	1. L’application affiche les commentaires d’une Mi et un bouton “Signaler” pour chaque commentaire. 2. L’utilisateur appuie sur le bouton de signalement. 3. L’application ouvre un pop-up avec deux entrées de textes : une pour le type de signalement et une autre pour un commentaire expliquant le signalement ainsi qu’un bouton “Envoyer”. 4. L’utilisateur remplit le type de signalement et peut choisir de remplir un commentaire. Il termine par appuyer sur le bouton “Envoyer”. 5. L’application affiche un nouveau pop-up remerciant l’utilisateur pour son signalement et envoie une notification de signalement au modérateur.

Scénario principal	UC3.7 Supprimer ses commentaires
Description	1. L’application affiche les commentaires d’une MI. 2. L’utilisateur trouve son commentaire et appuie sur l'icône “•••”. 3. L’application affiche le bouton “Supprimer”. 4. L'utilisateur appuie sur le bouton “Supprimer”. 5. L’application affiche un pop-up demandant la confirmation de la suppression du commentaire avec un bouton “annuler” et un “confirmer”. 6. L’utilisateur appuie sur bouton “confirmer”. 7. L’application supprime le commentaire de la liste des commentaires de la MI et envoie une notification à l'utilisateur l’informant de la bonne suppression du commentaire.

Scénario principal	UC3.8 Modifier ses commentaires
Description	1. L’application affiche les commentaires d’une MI. 2. L’utilisateur appuie sur l'icône “•••” puis sur le bouton “Modifier”. 3. L’application affiche le commentaire en mode édition.

	4. L’utilisateur change la note attribuée à la maison et/ou son commentaire puis il appuie sur le bouton “confirmer”. 5. L’application procède à la mise à jour du commentaire.

Scénario principal	UC3.9 Recherche un utilisateur
Description	1. L’utilisateur clique sur le bouton du menu des relations. 2. L’application affiche ses amis, les derniers commentaires de ceux-ci et un bouton “Ajouter un ami”. 3. L’utilisateur appuie sur le bouton “Ajouter un ami”. 4. L’application affiche une barre de recherche demandant d’entrer le nom de l’ami à ajouter et un bouton “Envoyer”. 5. L’utilisateur remplit la barre.

Scénario principal	UC3.10 Refuser une demande d’ami
Description	1. L’application envoie à l'utilisateur une notification l’informant qu'il a reçu une demande d’ami. 2. L’utilisateur ouvre ses notifications. 3. L’application informe l’utilisateur que tel autre utilisateur le demande en ami et affiche un bouton “Accepter” et un autre “Refuser” . 4. L’utilisateur appuie sur le bouton “Refuser”.

Scénario principal	UC4.1 Proposer une nouvelle maison d’illustre
Description	1. L’utilisateur clique sur le bouton “Je propose”. 2. L’application ouvre un pop-up affichant plusieurs entrées de texte dont : le nom de la maison, le nom de l’illustre auquel elle est attachée, son adresse, et une explication du choix de cet ajout ainsi qu’un bouton “Envoyer”. 3. L’utilisateur saisit les informations et appuie sur le bouton “Envoyer”. 4. L’application affiche un pop-up remerciant l’utilisateur de sa proposition et envoie celle-ci aux administrateurs.

Scénario principal	UC4.2 Ajouter une MI aux favoris
Description	1. L’utilisateur appuie sur une MI. 2. L’application affiche une prévisualisation de la maison (nom, photo, note, favoris, courte description, bouton “Commenter” et bouton “Plus d’info”). 3. L’utilisateur appuie sur l'icône “♥”.

	4. L’application ajoute la MI à la liste des favoris de l’utilisateur.

Scénario principal	UC4.3 Recommander une MI a un ami
Description	1. L’application affiche les informations de la MI. 2. L’utilisateur appuie sur le bouton “•••” puis sur “Recommander”. 3. L’application met cette recommandation en valeur dans les barres de recherche et filtres des amis de l’utilisateur.

Scénario principal	UC4.4 Modifier les informations d’une MI
Description	1. Le responsable MI appuie sur le bouton “Ma Maison d’illustre”. 2. L’application affiche les informations de la MI. 3. Le responsable MI appuie sur l’icon “•••” puis sur “Modifier”. 4. L’application passe en mode modification et affiche un formulaire pré-rempli avec les informations de la maison ainsi que deux boutons “Annuler” et “Enregistrer”. 5. Le responsable modifie les informations souhaitées puis appuie sur le bouton “Enregistrer”. 6. L’application enregistre les modifications et met à jour les informations puis envoie une notification au responsable.

Scénario principal	UC5.1 Faire une donation à une MI
Description	1. L’application affiche les informations de la MI. 2. L’utilisateur appuie sur le bouton “Faire une donation”. 3. L’application ouvre un pop-up avec les choix de paiements. 4. L’utilisateur appuie sur son moyen de paiement, entre ses coordonnées bancaires et confirme son paiement. 5. L’application affiche un pop-up remerciant le donateur.

Scénarios alternatifs

Scénario alternatif	UC1.1 Rechercher une maison d'illustre: Recherche inconnue
Description	1. L'utilisateur entre un nom de maison, un nom d'illustre ou une œuvre de l'illustre qui n'est pas connu de l'application dans la barre de recherche. 2. L'application remplace le texte de la barre de recherche par un message indiquant qu'aucune maison ne correspond pas à la recherche.

Scénario alternatif	UC2.1 Filtrer les maisons d'illustres selon des critères : Filtres qui ne correspondent à aucune maison
Description	1. L'utilisateur clique sur le bouton des filtres. 2. L'application affiche une choice box des filtres disponibles. 3. L'utilisateur appuie sur la choice box "Filtrer par". Il choisit ses filtres selon l'époque de l'illustre, une localisation précise, les notes des maisons, les recommandations de ses amis et ses favoris puis il valide ses filtres. 4. L'application ne trouve pas de maisons réunissant ces filtres. 5. L'application affiche un message indiquant qu'aucune maison ne correspond à ces critères.

Scénario alternatif	UC2.2 Commenter une maison d'illustre : Pas de texte
Description	1. L'utilisateur clique sur la maison pour laquelle il veut écrire un commentaire. 2. L'application affiche les informations sur la maison. 3. L'utilisateur appuie sur le bouton "Commenter". 4. L'application ouvre un champ d'écriture. 5. L'utilisateur ne renseigne que la note de l'application. 6. L'utilisateur appuie sur le bouton pour poster son commentaire. 7. L'application renvoie une erreur indiquant qu'il faut ajouter du texte au commentaire.

Scénario alternatif	UC2.2 Commenter une maison d'illustre : Pas de note
Description	1. L'utilisateur clique sur la maison pour laquelle il veut écrire un commentaire. 2. L'application affiche les informations sur la maison. 3. L'utilisateur appuie sur le bouton "Commenter". 4. L'application ouvre un champ d'écriture.
	5. L'utilisateur ne renseigne que le contenu texte du commentaire. 6. L'utilisateur appuie sur le bouton pour poster son commentaire. 7. L'application renvoie une erreur indiquant qu'il faut ajouter une note au commentaire.

scénario alternatif	UC2.5 Modifier ses information de compte : Nouveau pseudo incorrect
Description	1. L'utilisateur clique sur la partie relation de l'application puis sur le bouton "Modifier" en face de son pseudonyme. 2. L'application ouvre une nouvelle page qui affiche son profil en mode modification et affiche ses informations dans des champs de texte. 3. L'utilisateur renseigne un pseudo vide ou déjà existant et appuie sur le bouton "Enregistrer". 4. L'application renvoie une erreur indiquant que le pseudo ne peut être vide ou qu'il existe déjà.

Scénario alternatif	UC2.5 Modifier ses information de compte : Nouvelle adresse mail incorrecte
Description	1. L'utilisateur clique sur la partie relation de l'application puis sur le bouton "Modifier" en face de son pseudonyme. 2. L'application ouvre une nouvelle page qui affiche son profil en mode modification et affiche ses informations dans des champs de texte. 3. L'utilisateur renseigne une adresse mail vide ou déjà existante et appuie sur le bouton "confirmer la modification "Enregistrer". 4. L'application renvoie une erreur indiquant que l'adresse mail ne peut être vide ou qu'elle existe déjà.

Scénario alternatif	UC3.4 Accepter un ami : La demande n'est pas traitée
Description	1. L'application envoie à l'utilisateur une notification l'informant qu'il a reçu une demande d'ami. 2. L'utilisateur ouvre ses notifications. 3. L'application informe l'utilisateur que tel autre utilisateur le demande en ami et affiche un bouton "Accepter" et un autre "Refuser". 4. L'utilisateur ignore la notification. 5. L'application envoie une notification au bout de 2 semaines.

Scénario alternatif	UC3.8 Modifier ses commentaires : Contenu texte vide

Description	1. L'application affiche les commentaires d'une MI. 2. L'utilisateur appuie sur l'icône "•••" puis sur le bouton "Modifier". 3. L'application affiche le commentaire en mode édition. 4. L'utilisateur change son commentaire par un contenu vide puis il appuie sur le bouton "Confirmer". 5. L'application renvoie une erreur indiquant que le contenu texte ne peut être vide.

Scénario alternatif	UC3.9 Recherche un utilisateur : Utilisateur inconnu
Description	1. L'utilisateur clique sur le bouton du menu des relations. 2. L'application affiche ses amis, les derniers commentaires de ceux-ci et un bouton "Ajouter un ami". 3. L'utilisateur appuie sur le bouton "Ajouter un ami". 4. L'application affiche une barre de recherche demandant d'entrer le nom de l'ami à ajouter et un bouton "Envoyer". 5. L'utilisateur remplit la barre avec un nom inconnu. 6. L'application informe l'utilisateur que l'utilisateur est inconnu.

Scénario principal	UC3.10 Refuser une demande d'ami : Demande non-traité
Description	1. L'application envoie à l'utilisateur une notification l'informant qu'il a reçu une demande d'ami. 2. L'utilisateur ouvre ses notifications. 3. L'application informe l'utilisateur que tel autre utilisateur le demande en ami et affiche un bouton "Accepter" et un autre "Refuser". 4. L'utilisateur ignore la notification. 5. L'application envoie une nouvelle notification après 2 semaines.

Scénario alternatif	UC4.1 Proposer une nouvelle maison d'illustre : Champs vides
Description	1. L'utilisateur clique sur le bouton "proposer une MI". 2. L'application ouvre un pop-up affichant plusieurs entrées de texte dont : le nom de la maison, le nom de l'illustre auquel elle est attachée, l'adresse et une explication du choix de cet ajout ainsi qu'un bouton "Envoyer". 3. L'utilisateur saisit les informations, laisse un ou plusieurs champs vides et appuie sur le bouton "Envoyer".
	4. L'application informe l'utilisateur qu'il faut remplir les champs vides.

Nom du scénario alternatif	UC4.4 Modifier les informations d'une MI : Champs vides
Description	1. Le responsable MI appuie sur le bouton "Ma Maison d'illustre". 2. L'application affiche les informations de la MI. 3. Le responsable MI appuie sur l'icon "•••" puis sur "Modifier". 4. L'application passe en mode modification et affiche un formulaire pré-rempli avec les informations de la maison ainsi que deux boutons "Annuler" et "Enregistrer". 5. Le responsable modifie les informations souhaitées avec des informations erronées puis appuie sur le bouton "Enregistrer". 6. L'application informe l'utilisateur qu'il faut remplir les champs vides.

Scénario alternatif	UC5.1 Faire une donation à une MI : Informations erronées
Description	1. L'application affiche les informations de la MI. 2. L'utilisateur appuie sur le l'icon "•••" puis "Faire une donation". 3. L'application ouvre un pop-up avec les choix de paiements. 4. L'utilisateur appuie sur son moyen de paiement, et entre des informations erronées. 5. L'application informe l'utilisateur des informations incorrectes.

Schéma Logique Relationnel

Region(idRegion,regionName)
Departement(idDepartement,#idRegion,departmentName)
Outstanding(idOutstanding,outstandingName,type,epoque)
Work(idWork,#idCreator,nameWork)
House(idHouse, #idOutstanding, #idDepartement, #idRegion, houseName, presentation, city,adresse,postalcode,latitude,longitude,websiteLink,instaLink,facebooklink,twitterLink,foundingDate,labelDate)
Img((idImg, #idHouse),url,name)
User(idUser,#manage,password,blame,email,login,isModerate)
Friend((#idUser1,#idUser2))
Favorite((#idUser,#idHouse))
Notification(idNotif,#idOwner,#requestfriend,#reportComment,message)
Comment(idCom,#idOwner,#idHouse,content,rate,reporting,like,dislike)
ProhibitedEmail(mail)
ProhibitedLogin(login)

Fichier create.sql de la DB

```
-- Region(idRegion,regionName)
CREATE TABLE Region (
    idRegion int primary key,
    regionName varchar(255)
)
-- Departement(idDepartement,#idRegion,departmentName)
CREATE TABLE Department (
    idDepartment int primary key,
    idRegion int references region(idRegion),
    departmentName varchar(255)
)
-- Outstanding(idOutstanding,outstandingName,type,epoque)
CREATE TABLE Outstanding (
    idOutstanding int primary key,
    outstandingName varchar(255),
    type varchar(255),
    epoque varchar(255)
)
--House(idHouse, #idOutstanding, #idDepartement, #idRegion, houseName,
presentation, city, adresse, postalcode, latitude, longitude, websiteLink,
instaLink, facebooklink, twitterLink, foundingDate, labelDate)
CREATE TABLE House (
    idHouse int primary,
    idOutstanding int references outstanding(idOutstanding),
    idDepartment int references Department(idDepartment),
    idRegion int references Department(idRegion),
```

```
    houseName varchar(255),
    presentation varchar(255),
    city varchar(255),
    adress varchar(255),
    postalcode int,
    latitude varchar(255),
    longitude varchar(255),
    websiteLink varchar(255),
    instaLink varchar(255),
    facebookLink varchar(255),
    twitterLink varchar(255)
    foundingDate int,
    labelDate int
)
-- Img((idImg, #idHouse),url,name)
CREATE TABLE Img (
    idImg int primary,
    idHouse int references House(idHouse) primary,
    url varchar(255),
    imgName varchar(255)
)
-- User(idUser,#manage,password,blame,email,login,isModerate
CREATE TABLE User (
    idUser int primary,
    manage int references House(idHouse),
    password varchar(255),
    blame int,
    email varchar(255) UNIQUE,
    login varchar(255) UNIQUE,
    isModerate BOOLEAN
)
-- Friend((#idUser1,#idUser2))
CREATE TABLE Friend (
    idUser1 int primary references User(idUser),
    idUser2 int primary references User(idUser)
)
-- Favorite((#idUser,#idHouse))
CREATE TABLE Favorite(
    idOwner int references User(idUser) primary,
    idHouse int references House(idHouse) primary
)
-- Notification(idNotif,#idOwner,#requestfriend,message)
CREATE TABLE Notification (
    idNotif int primary,
    idOwner int references User(idUser),
    requestfriend int references User(idUser),
```

```
        reportComment int reference Comment(idCom),
        message varchar(255)
)
-- Comment(idCom,#idOwner,#idHouse,content,rate,reporting,like,dislike)
CREATE TABLE Comment (
    idCom int primary,
    idOwner int references User(idUser) UNIQUE,
    idHouse int references House(idHouse),
    content varchar(255),
    rate int,
    reporting int,
    like int,
    dislike int
)
-- ProhibitedEmail(mail)
CREATE TABLE ProhibitedEmail (
    mail varchar(255) primary
)
-- ProhibitedLogin(login)
CREATE TABLE ProhibitedLogin (
    login varchar(255) primary
)
```

Diagrammes d'objets

Ce premier diagramme d'objets permet de comprendre le fonctionnement des classes filles de Person et de la classe Notification.

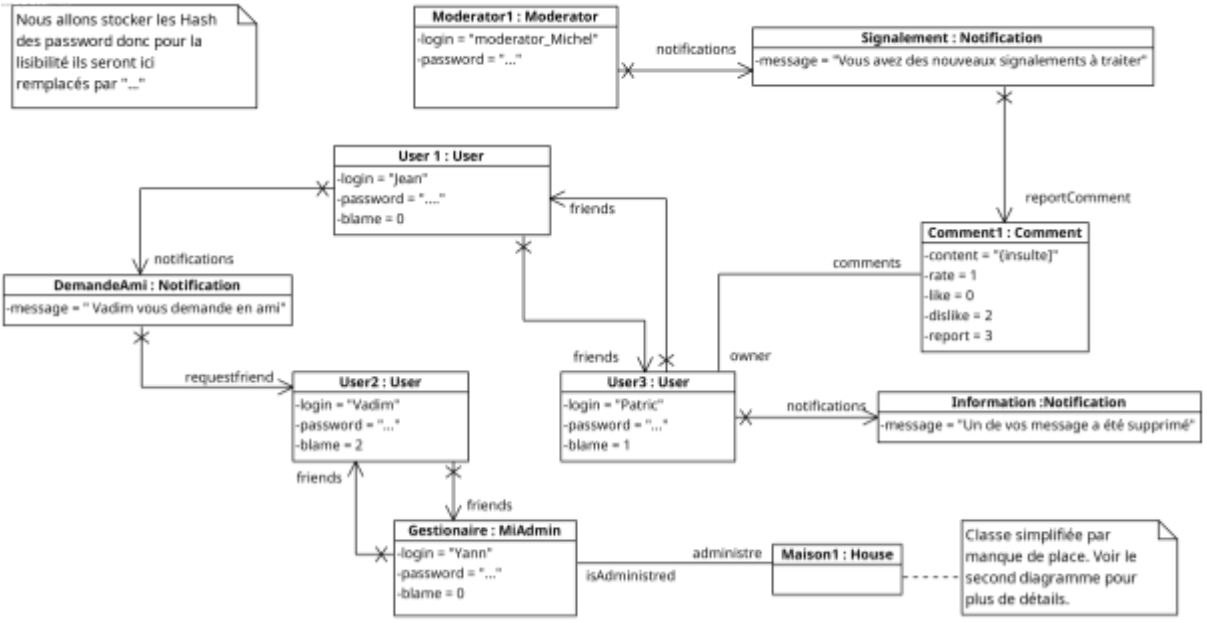


Diagramme d'objets - Person

Ce deuxième Diagramme décrit le fonctionnement de la classe Comment.

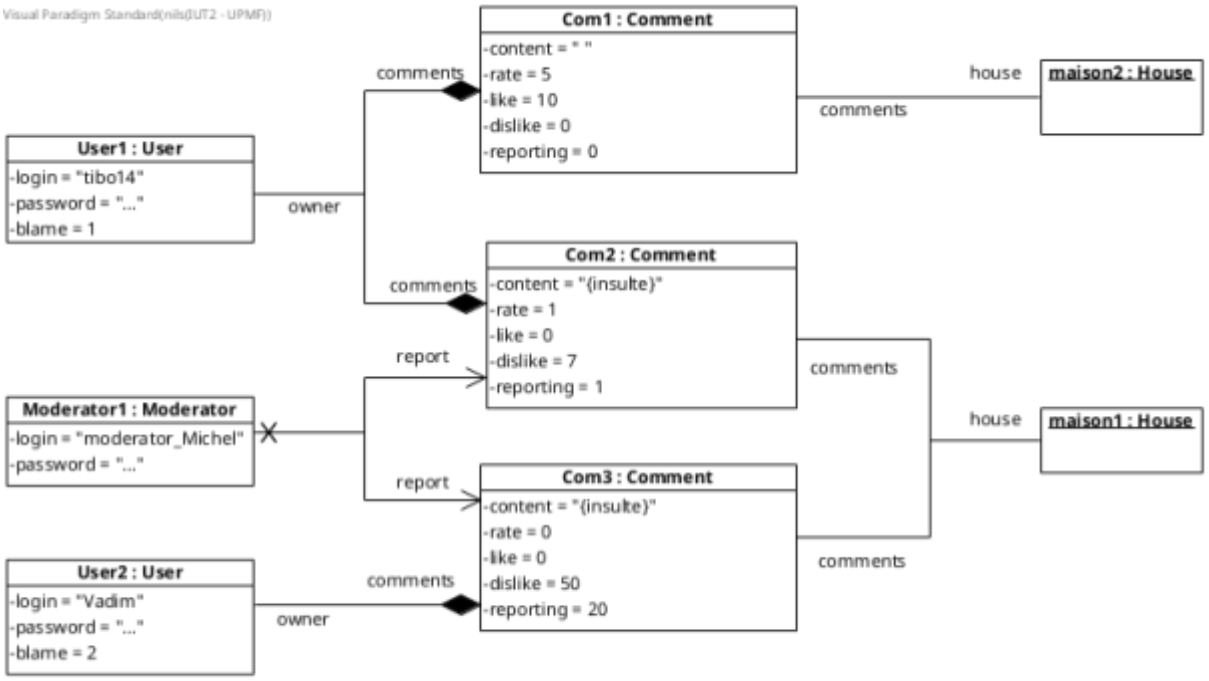


Diagramme d'objets - Comment

Ce dernier diagramme présente le fonctionnement de la Classe House.

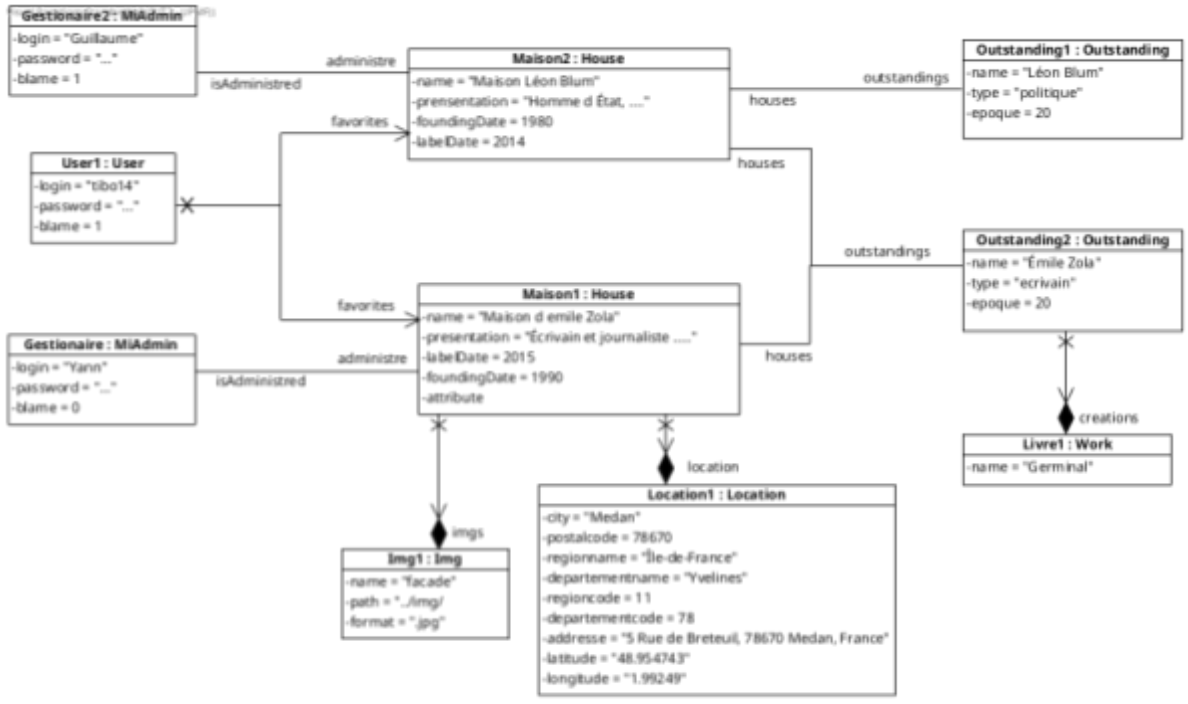


Diagramme d'objets - House

Evaluation des interfaces - Balade Cognitive

Lors de la balade cognitive réalisée sur nos maquettes, nous avons pu remarquer certains problèmes sur nos interfaces.

Dans un premier temps, le guidage n'était pas assez explicite. En effet, lors de la réalisation de la fonctionnalité de filtrage des maisons, le bouton destiné à cet effet n'était pas assez mis en avant. Cela a conduit le testeur à vouloir entrer les critères de filtrage directement dans la barre de recherche de l'application.

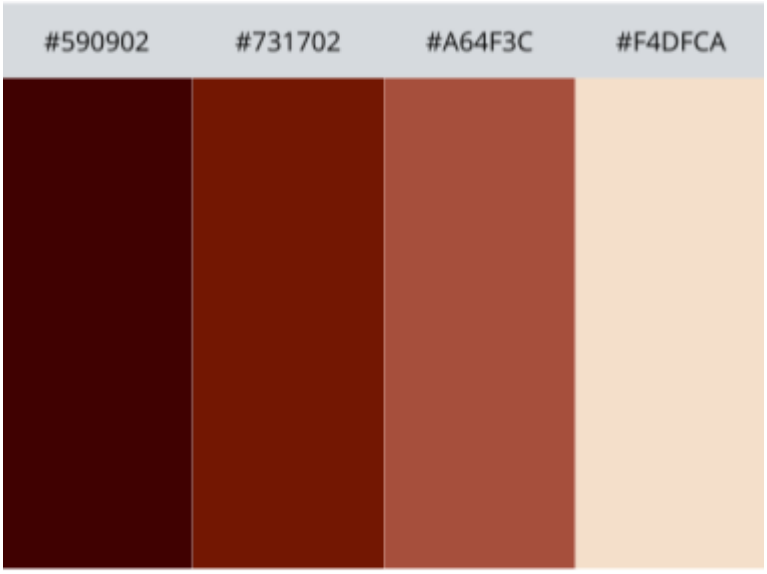
De plus, l'utilisateur a rencontré un problème pour accéder à ses amis. En effet, son premier réflexe a été de cliquer sur l'icône utilisateur en haut à droite de l'écran. C'est pourquoi, nous avons déplacé les informations utilisateur au-dessus des amis. Cela permet de mieux respecter la signifiance des codes et dénominations puisque l'icône utilisé pour les amis représente aussi le compte de l'utilisateur dans beaucoup d'applications.

Charte graphique et logo

Pour notre application, nous avons décidé d'utiliser principalement des teintes de marron pour rappeler les différents éléments des bâtiments en bois. Dans ces teintes il y a la couleur des briques et des tuiles d'une maison, un marron ressemblant davantage au bois, et du beige. Cette gamme est souvent associée à la stabilité, la solidité (surtout au travers du bois) et la fiabilité.

Pour le logo ci-dessous, nous avons choisi de mettre en valeur la France. On retrouve ainsi le drapeau tricolore sur toute la largeur du pictogramme de carte. L'icône de punaise reprend une des couleurs principales de l'application : le beige. De manière générale, le logo associe les notions de cartes interactive et de maisons d'illustres grâce à l'icône de maison au centre de la punaise.

Logo de l'application



Charte graphique

#590902 #731702 #A64F3C #F4DFCA